International Journal of Petroleum and Geoscience Engineering (IJPGE) 2 (2): 138-150, 2014
ISSN 2289-4713

Research paper

Prediction of Maximum Oil Production by Gas Lift in an Iranian Field Using Auto-Designed Neural Network

E. Khamehchi [a,*], H. Abdolhosseini [a], R. Abbaspour [a]

[a] Faculty of Petroleum Engineering, Amir Kabir University of Technology, Tehran, Iran

* Corresponding author.
E-mail address: khamehchi@aut.ac.ir

A b s t r a c t

Keywords:

Gas Lift,
Neural Network,
Genetic Algorithm,
Optimization.

Artificial neural networks have been becoming increasingly popular in oil industry over the last decades. But there was not a specific framework and procedure to design appropriate networks in respect to the problem. One of drawbacks of neural network application is its dependence on designer's experience. In this work we proposed a method in which we design an artificial neural network coupling it with a genetic algorithm to not only optimize weights and biases but also number of neurons and connections. This method can be used to design complex systems in which time and simplicity are important factors as we used it in predicting gas lift aided recovery to obviate the need to run simulation software which is expensive and time consuming. First we create a network with neural network toolbox of MATLAB. This network was built fully-connected. Then we start our program with this initial guess and compare the final structure and mean square error (MSE) with the network created by MATLAB. The network obtained by our program was simpler and also it has lower MSE indicating a network that is simpler and more accurate.

Accepted:02 June2014

1. Introduction

In 1956, ten scientists in a conference at Dartmouth C. delineated what today is called artificial intelligence [1]. Since 1980, AI has become commercial. Expert systems have proliferated. Applications in computers, electronics, oil, medicine, and the military have all become well-known. Artificial neural networks, fuzzy logic, and evolutionary algorithms are common among AI techniques being applied today in oil and gas industry. AI has a lot of applications in petroleum industry like prediction of reservoir properties [2], well log correlation [3], reservoir characterization

[4], history matching [5], production forecast [6], drilling operations [7], EOR [8], PVT properties estimation [9], well test interpretation [10], gas lift optimization [11,12] and etc..

One of the application of AI in petroleum industry is predicting and optimizing gas lift operations. Using genetic algorithm for optimizing a gas lift process requires application of commercial software to simulate the process in each step and for every chromosome. Due to long time required for running the software and high cost, in this work we try to create a neural network to predict gas lift operation performance to obviate the need to use software.

In the following sections, we first explain the basic concepts of gas lift, genetic algorithms and neural network. Then we use a method to optimize the neural network with genetic algorithm to obtain smaller and optimum network with an acceptable accuracy. Then we construct an optimum neural network for predicting the behaviour of a gas lift operation on an Iranian oil field and compare it with a fully connected neural network created by MATLAB.

2. Gas lift

The energy which drives oil from underground reservoirs to the surface generally comes from five main mechanisms: water drive, solution gas drive, gas cap drive, gravity segregation and a combination of these mechanisms. When this natural energy becomes inefficient for driving oil to surface or delivered liquid is not enough, this energy must be compensated by artificial methods. These methods are called artificial lift methods. Usually artificial method is used when the natural energy could drive the liquid to the wellbore, if not we must use secondary or EOR methods.

Artificial methods are basically try to decrease the pressure gradient in well and bottom hole flowing pressure. Artificial lift methods are divided into two main categories:

1- Gas lift
2- Lift using pumps

Gas lift is one of the most practical artificial lift methods in oil industry. In this method, using surface compressors and also valves which are placed on tubing, gas is injected to the fluid that enters the tubing and as a result of lightening the fluid, the hydrostatic pressure is decreased.

Gas lift uses an external source of high pressure gas for supplementing formation gas to lift the well fluids. Gas lift is particularly applicable to lifting wells where high-pressure gas is available. Gas compressors may have been installed for gas injection, or high pressure gas wells may be nearby. Since the cost of compression far exceeds the cost of down-hole gas lift equipment, gas lift always should be considered when an adequate volume of high-pressure gas is available for wells requiring artificial lift [13].

Gas lift has advantages over pump method since it can be used in wells with sand production, wells with high depth and sea wells and with respect to pumps it has less depth limitations. On the other hand, gas lift is only used in large fields where lots of wells exist and applying it in small fields is not economic. Generally, there are two main categories of gas lifting, continuous and intermittent.

Continuous: Continuously injecting gas into the tubing or casing at a predetermined depth to reduce the pressure opposite the producing formation.

Intermittent: Injection of high pressure gas into the tubing at sufficient volume and pressure to lift the fluid head accumulated above the valve with maximum velocity

Deciding which procedure to be used depends on a lot of parameters but generally the four well categories considered for gas lift are as follows;

1) High PI – High BHP continuous

2) High PI – Low BHP intermittent

3) Low PI – High BHP continuous

4) Low PI – Low BHP intermittent

Depth of injection point and injection rate are the most important parameters in designing a gas lift system. For determining the injection depth, first, we draw the pressure gradient curve in well and then the depth where this curve and gas gradient curve intersects is the injection depth. Note that we must reduce the pressure loss due to injection valves from gas gradient curve.

Optimum injection rate results in maximum oil production rate. To determine this rate for each injection rate, oil production rate is calculated and optimum gas injection rate is the rate that corresponds to the maximum oil production rate. This procedures is implemented using correlations or softwares like WELLFLO and PROSPER. These softwares produce the curve of oil production rate versus gas injection rates which is called GLPC. Fig. 1 shows a GLPC curve for an imaginary well.

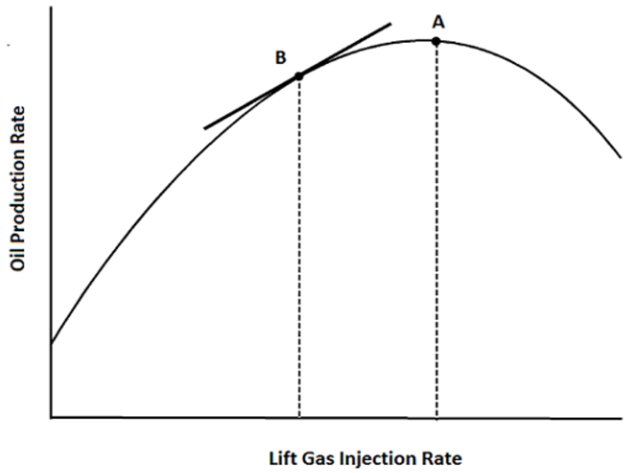

Fig.1: Optimum point (A) and economic point (B) in gas lift performance curve

If we consider a well under just tubing head pressure control and with unlimited gas supply increasing the gas injection rate first results in continuous increase in oil production rate due to reduced density of fluids in tubing and as a result pressure head in tubing decreased, but beyond a certain rate, oil production rate decreases due to noticeable increase in pressure losses due to friction. For an individual well with no constraints other than a tubing head pressure limit, the optimum gas lift injection rate is the value at the peak(point A). In reality due to economic and technical constraints we use another point near the optimum point like point B.

When gas supply is limited and the optimization must apply on a group of wells instead of an individual one, another problem called gas allocation occurs. In this case we must allocate gas to each well in a way that the total oil production becomes maximum.

In gas lift operation optimization, generally there is a function that needs to be minimized or maximized like cost function (minimization) or total oil production rate (maximization). Most important part in gas lift optimization is gas allocation problem. There are several optimization algorithms for this problem like: graphical algorithm, numerical algorithm and heuristically algorithms like genetic algorithm. The problem with the last algorithm is the need to predict the performance of wells in every optimization step. To eliminate this problem we could use neural network to predict parameters like production rate in every optimization step. In this work we present an auto-designed neural network to do this task and reduce the computational cost.

3. Genetic algorithm

Genetic algorithm (GA) is an optimization and search technique based on genetics and natural selection principles. A genetic algorithm allows a population composed of many individuals to evolve under specified selection rules to the state that maximizes the "fitness" (i.e., minimizes the cost function). John Holland developed genetic algorithm in his book [14]. This method became popular by one of his students, David Goldberg, who was able to solve a difficult problem involving the control of gas-pipeline transmission for his dissertation [15]. Holland was the first to try to develop a theoretical basis for GAs through his schema theorem. De Jong showed the usefulness of the GA for function optimization and made the first effort to find optimized GA parameters. [16]. Since then, many versions of evolutionary programming have been tried with varying degrees of success.

There are some primary structures in GA as follows:

Chromosome: Each possible solution for the optimization problem is called chromosome. Usually chromosomes are shown in binary format (string of bits) that composed of genes.

Population: An ensemble of chromosomes (possible solutions) is called population in genetic algorithm

Fitness function: In order to compare the solutions (chromosomes) in genetic algorithm we must define an appropriate fitness function to estimate the proximity of each solution to desired solution. In this way we could select more suitable answers in each step.

3.1. Genetic algorithm operators

By applying genetic algorithm operators in reproduction stage, new generation is created from past generation. Selection, crossover and mutation are the most common operators in genetic algorithm.

Selection: This operator selects a number of chromosomes from population for reproduction. There are several algorithms for selection in genetic algorithm like fittest selection, roulette wheel and tournament selection.

Crossover: In mutation a part of chromosomes exchange with each other to generate offsprings that can be fitter than their parents. In this procedure we randomly select a point in a chromosome and exchange the following genes with genes from another chromosome. Based on the number of crossover points we define single point crossover, two point crossover, multiple point crossover and uniform crossover. In Fig. 2 you can see a single point crossover.

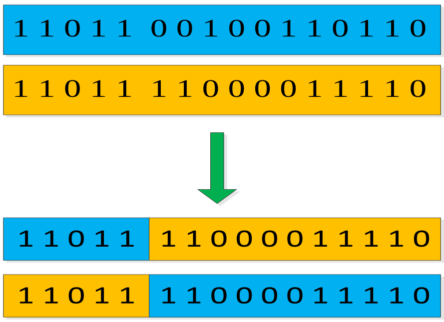

Fig.2: Single point crossover

Mutation: This operator is applied after crossover. Some of genes in chromosomes are randomly selected and changed to another value for example in binary system 0 becomes 1 and vice versa. After mutation new chromosomes create the new generation and this process goes on till the termination condition reached.

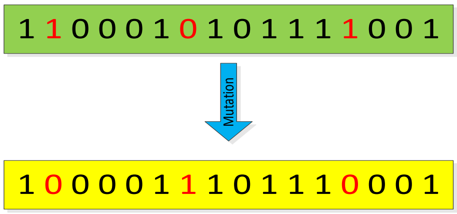

Fig.3: Mutation

Termination condition: GA is based on producing and testing solutions to reach the optimum condition. Since we do not know the answer in advance we cannot define the correct solution as the termination condition so we must use other criteria. There are several termination conditions for GA like:

1- A certain number of generations
2- Failure to improve the fitness of the best over several consecutive generations
3- Time limit

A combination of above can also be used.

GA should be used in problems when the search space for solution is large. GA in these spaces moves faster toward the solution compared to other algorithms

4. Neural networks

The class of neural networks (NNs) is a subclass of parallel distributed processing (PDP) models. These models assume that information processing is a result of interactions between simpler processing elements called nodes [17]. A neuron is a nonlinear, parameterized, bounded function. The variable of neuron is often called input of neuron and its value is its output. In Fig. 4 a simple neuron is shown.

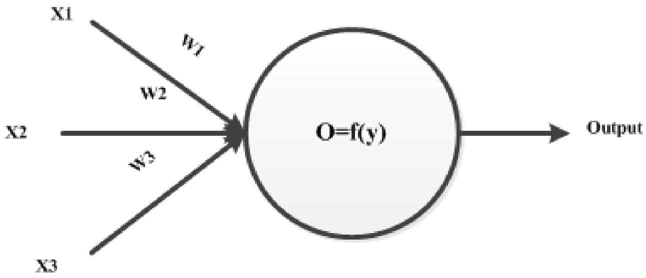

Fig.4: A simple neuron without bias

Where,

$$y = \sum_{i=1}^{n} x_i w_i$$

Each input of neuron (x) has a weight (w) and y is a weighted sum of inputs. f is a transfer function. Linear and sigmoid functions are two of the most popular transfer functions in neural network. A combination of these neurons creates a neural network. Neural network structure is consist of an input layer, an output layer and several intermediate layers called hidden layers which can have any desired number of neurons. Neural networks come in two classes: feedforward and feedback or recurrent networks. A feedforward neural network is a set of neurons connected together, in which the information flows only in the forward direction, from inputs to outputs. In recurrent neural network(RNN), there is at least one path that, following the connections, leads back to the starting neuron; such path is called a cycle.

4.1. Training of Neural Networks

Training is the algorithmic procedure whereby the parameters of the neurons of the network are estimated. There are two training categories: Supervised Training and Unsupervised Training.

There are useful resources for basic principles of neural network; books [18,19]; and articles [20]; [21].

4.2. Auto-Design of Neural Networks

Two steps in neural network design are structure design of network and selecting the training method and parameters. In structure design, we determine number of neurons, layers and transfer function between layers and in training method selection, we generally determine weights and biases.

Generally, there are three models of structural learning: constructive algorithms, destructive or pruning algorithms, and parallel algorithms. Constructive algorithms start with a small network and train this network until it is unable to continue learning, then new components are added to the network. This process is repeated until a satisfactory solution is reached [22]. Destructive methods also known as pruning algorithms, start with a big network, that is able to learn but usually over-fits the training data, and try to remove the connections and nodes that are not useful [23]. A major problem with pruning method is the assignment of credit to structural components of the network in order to decide whether a connection or node must be removed. Constructive method has several advantages over destructive method [22]:

- Initial network is easy to specify
- Small solutions are searched first and preferred if bigger ones do not improve the performance
- They are faster as small networks are tested first

A disadvantage of constructive method is that it may reach suboptimal solution.

In parallel methods the performance of a set of neural networks are evaluated simultaneously and based on updating rules a new set of neural networks are obtained. This cycle is repeated until convergence is achieved. Almost all parallel methods are based on evolutionary algorithms like GA. There have been several motivations for evolving neural networks. Often, designing a neural network requires knowledge of the problem domain. But in many real applications such knowledge is unavailable, which usually leads to a repetitious trial-and-error approach. Back propagation and other gradient descent algorithms are used to find a global minimum in an error space, but they may get stuck in local minimum, and require error space to be differentiable. However, evolutionary algorithms do not require gradient information so that they can search virtually any kind of error space. In design of neural network structure, parallel methods are divided into two groups based on their encoding mechanisms: direct methods and indirect methods. Two important aspects of using a GA algorithm are setting an encoding mechanism and a fitness function. As in neural network structure optimization all variables are discrete, a binary mechanism is appropriate. The difference between the direct and indirect method is in representing the information of the structure in binary strings. Direct method as it sounds directly represents all information in the binary string but indirect methods use an interpretation method to obtain parameters from binary strings. In indirect method data is compressed to reduce the length of strings which is an advantage over the direct method. The difference between indirect methods is in compression and interpretation procedures.

5. Methodology

We use a method for auto-design of neural networks which is recent in oil industry called Genetic Auto-Design of Neural Network (GADONN) [24].This method is an indirect parallel method. GADONN uses a context-free L-system [25,26] to encode the developmental rules in genotype. Restricting the search domain to general feedforward networks will lead to a lower triangular matrix [24]. The information about connection between neurons and signal weight and bias of each neuron is represented in a chromosome. In order to convert connections between neurons, first we construct connectivity matrix like Fig. 5.

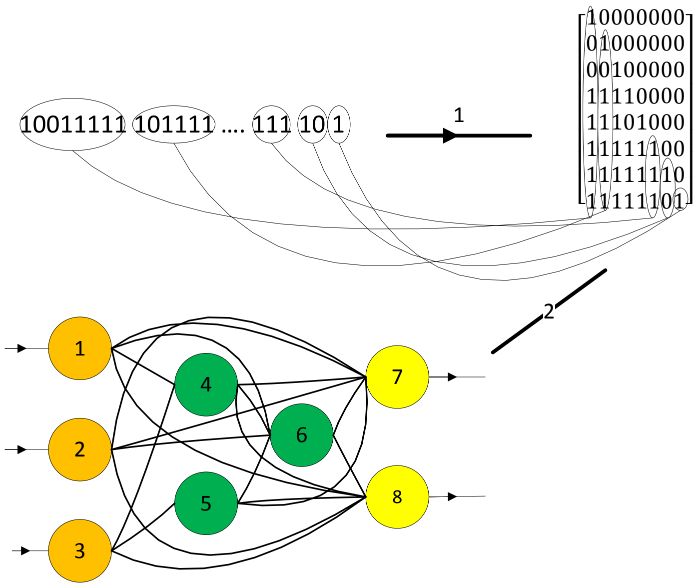

Fig.5: Different stage of decoding of binary string

Implementing the procedure illustrated in Fig. 5, we are able to convert the network to the form of a matrix and then transform it to a vector. This vector would be the input for GA. For a neural network with N neurons, the length of the vector would be $N(N+1)/2$ which means first $N(N+1)/2$ genes of chromosome are from connectivity matrix. This part of chromosome is binary and as a result only takes 0 and 1 as its value.

The other two parameters which are represented in chromosomes are weight of each signal and corresponding bias of each neuron. The number of weight parameters is $N(N-1)/2$ and for bias this is N and sum is $N(N+1)/2$. Unlike the first $N(N+1)/2$ genes, values of these genes are continuous. For handling this problem we decided to convert these continuous values to 0 and 1 form. We assume that these parameters vary between 0 and 2 and with 4bits resolution we convert them to a 0 and 1 form. This assumption is used to reduce the run time of program which may result in some degree of error but it can be modified if needed. As a result the final length of the chromosome will be$5 \times N(N+1)/2$.

Another parameter that can be changes is transfer function. In this work, the *TANSIG* function is used for hidden layers and *PURELIN* is used for all of the output neurons.

In every GA run, new chromosomes are generated. These binary chromosomes contain the network parameters. The last $N(N-1)/2$ parameters are representation of signal weights and biases of each neuron, by using a function called *b2doub* we convert this valued to a double form and the output of this function will be a vector of the length $N(N-1)/2$. Then we call another function, *convert2mat*, which converts the vector to a matrix. In this new matrix, values on the main diagonal will be biases and values below the main diagonal will be signal weights.

The first $N(N-1)/2$ genes also convert to a matrix using the same function. The resulting matrix is the new connectivity matrix (cm). This matrix has a special form that is similar to the matrix in Fig.6.

$$
Cm = \begin{bmatrix}
\begin{smallmatrix}1&0&\cdots&0\\0&1&\dots&0\\\vdots&\vdots&\ddots&\vdots\\0&0&\cdots&1\end{smallmatrix} & & & & \\
\begin{smallmatrix}1&\cdots&1\\\vdots&\ddots&\vdots\\1&\cdots&1\end{smallmatrix} & \begin{smallmatrix}1&0&\cdots&0\\0&1&\dots&0\\\vdots&\vdots&\ddots&\vdots\\0&0&\cdots&1\end{smallmatrix} & & & \\
\cdots \quad \cdots & \cdots \quad \cdots & \cdots & \cdots & \\
\ddots & \ddots & \ddots & & \\
\cdots \quad \cdots & \cdots \quad \cdots & \cdots & \cdots & \\
\begin{smallmatrix}1&\cdots&1\\\vdots&\ddots&\vdots\\1&\cdots&1\end{smallmatrix} & \cdots \quad \ddots \quad \cdots & \cdots \quad \ddots \quad \cdots & \cdots & \begin{smallmatrix}1&0&\cdots&0\\0&1&\dots&0\\\vdots&\vdots&\ddots&\vdots\\0&0&\cdots&1\end{smallmatrix}
\end{bmatrix}
$$

Fig.6: General form of a connectivity matrix

The main diagonal is formed of a series of identity matrices. Elements above the main diagonal are zero and the elements below the main diagonal are zero and one. One is representing a connection and zero is representing lack of connection.

The output that GA gives each time needs a pre-processing. First, based on the primary network (first guess), connectivity matrix must be in the form of Fig. 6 in which elements of main diagonal are identity matrices. The other matter is that there may be a neuron that has only input without output or only output without input. This neuron is unneeded and must be deleted from the network. In the connectivity matrix if in a row or column there is only one nonzero element, that neuron is unneeded and must be deleted. As a result of deleting a neuron in the connectivity matrix, its corresponding row and column in weight and bias matrix must be equal to zero.

After these modifications, the network output is obtained. As a result of assuming the network fully-connected, input signals to the output layer can be from any neuron in previous layers, so, it is essential to compute the outputs of hidden layers before computing the last layer output.

The other important issue is defining the target function that GA is trying to minimize. This function must include the error of the network and topology of network including number of neurons and connections between them. The proposed target function is presented here

$$f = -k_1 MSE - k_2 N$$

Where, k_1, weight factor of network error; k_2, weight factor of number of neurons; MSE is the mean square error of network and N is the number of neurons. We could set the importance of MSE and number of neuron by selecting different values for k_1 and k_2.

6. Application

As presented before optimizing a gas lift process requires simulating the performance of wells in each step which takes a long time. In order to reduce the need to run simulators, we could use a neural network to predict required factors. In this work we tried to automatically design and optimize a neural network with genetic algorithm.

Our case study is an Iranian oil field with four potential wells for gas lift operation. Our goal is to design a neural network that can predict the performance of gas lift operation in order to obviate the need for commercial software which is expensive and time consuming.

Due to shortage of data available, we used design of experiment (DOE) method to generate some imaginary wells on which gas lift operation are performed. First we gathered data from 4 wells which were candidates for gas lift then we used Taguchi L64 methods to generate new data. The used parameters include: 3 reservoir parameters and 7 well parameters each one in two level. You can see the variables in Table. 1. We obtained these parameters by a sensitivity analysis and find that some parameters like GOR have little effect on optimum gas injection rate as shown in Fig. 7.

Table.1: Parameters and levels used in Taguchi method

Level	Parameters
20 – 25	API
2 – 3	Productivity Index(Stb/Psi.Day)
4000 – 4800	Bottom hole Static Pressure(Psi)
30 – 40	Water Cut (%)
3.240 – 3.958	Tubing Inside Diameter(inch)
5 – 6.538	Wellbore Size(inch)
10500 – 12000	Mid Perforation Depth(ft)
420 – 600	Flowing Wellhead Pressure(Psi)
8000 – 12000	Injection Depth(ft)
1 – 3.5	Gas Injection Rate(MM Scf/Day)

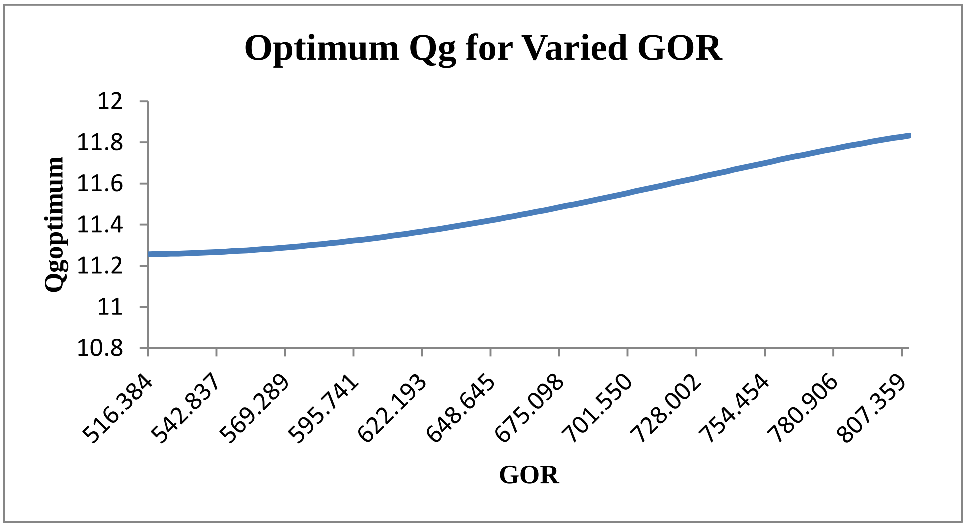

Fig.7: Effect of GOR on optimum gas injection rate

Using these parameters in 2 levels we obtained 64 different states for combination of these four wells. Each well was simulated by advance production software, *PROSPER*, and gas lift performance curve (GLPC) was plotted for each well. After simulation we get flowing bottom hole pressure of each well as the 8th well parameter. As a result we have $3 + 8 \times 4 = 35$ inputs. Maximum point of performance curve was taken as optimum gas injection rate. By adding up corresponding oil rate at optimum gas injection rates for 4 wells we obtain total maximum oil production rate which is our output parameter.

So, we start the program with a feedforward network with 35 inputs and 1 output (maximum oil production rate) and two hidden layers having 6 and 4 neurons respectively. The result of the program and a comparison with a fully-connected feedforward network constructed by MATLAB are illustrated in Table. 2.

Table.2: comparison between MATLAB neural network and designed network

	No. of Neurons	No. of Hidden Layers	No. of Connections	MSE
MATLAB NN	46	2	238	6.6674e+04
Auto-designed NN	44	2	94	86.22361

Using this proposed network we can easily use a genetic algorithm to optimize the gas lift operation without the need to use software to simulate the performance in every step which reduces the time and computational costs.

7. Conclusion

We used a recent method in oil industry to auto-design of neural networks. As the result of this program, we can obtain simpler structures for neural networks with the acceptable accuracy compared to the original network. This program not only creates a simpler network but also optimizes the biases and weights of neurons.

In this method effect of designer's experience on creating network decreases and so it could be a more general and well-designed network compared to conventional methods. This method could use for complex systems where both simplicity and accuracy are required. The structure of the final network is strongly dependent on the coefficients in target function. If we increase k_2 program will tend to decrease number of neurons more than decreasing MSE and vice versa.

Using the proposed neural network design we can obviate the need for running simulators in every optimizations step and can predict the complex behaviour of the gas lift operation with acceptable accuracy and cost effective.

References

[1] Lideniro, A., Potential Application for Artificial Intelligence in the Petroleum Industry. Journal of Petroleum Technology 1991:**43**(11): 1306-1309.

[2] Ali, S.S., et al., Hydraulic Unit Estimation from Predicted Permeability and Porosity Using Artificial Intelligence Techniques, in 2013 North Africa Technical Conference & Exhibition. 2013, 2013, Society of Petroleum Engineers: InterContinental Citystar,Cairo, Egypt.

[3] Kuo, T.B. and R.A. Startzman, An Artificial Intelligence Approach To Well Log Correlation. The Log Analyst 1987: **28**(2).

[4] Anifowose, F.A., Artificial Intelligence Application in Reservoir Characterization and Modeling: Whitening the Black Box, in SPE Saudi Arabia section Young Professionals Technical Symposium. 2011, Society of Petroleum Engineers: Dhahran, Saudi Arabia.

[5] Al-Thuwaini, J., G. Zangl, and R.E. Phelps, Innovative Approach to Assist History Matching Using Artificial Intelligence, in Intelligent Energy Conference and Exhibition. 2006, Society of Petroleum Engineers: Amsterdam, The Netherlands.

[6] Boomer, R.J., Predicting Production Using a Neural Network (Artificial Intelligence Beats Human Intelligence), in Petroleum Computer Conference. 1995, 1995 Copyright 1995, Society of Petroleum Engineers, Inc.: Houston, Texas.

[7] Gola, G., et al., Improving management and control of drilling operations with artificial intelligence, in SPE Intelligent Energy International. 2012, Society of Petroleum Engineers: Utrecht, The Netherlands..

[8] Ibatullin, R.R., et al., Application and method based on artificial intelligence for selection of structures and screening of technologies for enhanced oil recovery, in SPE/DOE Improved Oil Recovery Symposium. 2002, Copyright 2002, Society of Petroleum Engineers Inc.: Tulsa, Oklahoma..

[9] Alarfaj, M.K., A. Abdulraheem, and Y.R. Busaleh, Estimating Dewpoint Pressure Using Artificial Intelligence, in SPE Saudi Arabia section Young Professionals Technical Symposium. 2012, Society of Petroleum Engineers: Dhahran, Saudi Arabia..

[10] Houze, O.P. and O.F. Allain, A Hybrid Artificial Intelligence Approach in Well Test Interpretation, in SPE Annual Technical Conference and Exhibition. 1992, 1992 Copyright 1992, Society of Petroleum Engineers Inc.: Washington, D.C.

[11] Khishavand, M., Khamehchi, E., Nonlinear Risk Optimization Approach for Gas Lift Allocation Optimization, Industrial & Engineering Chemistry Research, 2012; 51: 2637-2643

[12] Rashidi, F., Khamehchi, E., Rasouli, H., Oil Field Optimization Based on Gas Lift Optimization, Faculty of chemical engineering, Amir Kabir University of technology, 2010

[13] Winkler, H.W., Gas Lift (1987 PEH Chapter 5). 1987.

[14] Holland, J.H., in Adaptation in Natural and Artificial Systems. 1975, University of Michigan Press: Ann Arbor.

[15] Goldberg, D.E., Genetic Algorithms in Search, Optimization and Machine Learning. 1989, Reading, MA: Addison-Wesley..

[16] De Jong, K.A., Analysis of the Behaviour of a Class of Genetic Adaptive Systems, P.D. Dissertation, Editor. 1975, University of Michigan: Ann Arbor.

[17] Mc Duff Spears, W., Using Neural Networks and Genetic Algorithms as Heuristics for NP-Complete Problems, in Computer Science. 1989, George Mason University: Faibanks, Virginia.

[18] Dreyfus, G., Neural Networks, Metohodology and Application 2005, Germany: Springer.

[19] Peterson, C., Soderberg, B., A New Method for Mapping Optimization Problems onto Neural Networks. International Journal of Neural Systems, 1989..

[20] Wilson, S.S., Teaching Network Connectivity Using Simulated Annealing on a Massively Parallel Processor. Proceeding of the IEEE 1991; **79**: 559.

[21] Herault, L., Niez, J. J., Neural Network and Combinatorial Optimization, A Study of NP Complete Graph problems. Neural Networks. Avances and Applications, 1991.

[22] Kwok, T.Y., Yeung, D. Y.,, Constructive Algorithms for Structure Learning in Feedforward Neural Networks for Regression Problems. Neural Networks, 1997; **8**: 630-645.

[23] Sutton, R.S., Two Problems with Backpropagation and Other Steepest-Descent Learning Procedures for Networks. Preceedings of the Eighth Annual Conference of the Cognitive Science Society 1986: 823-831.

[24 Boozarjomehry, R.B., Svreek, W. Y., Automatic Design of Neural Network Structure. Computers and Chemical Engineering, 2000..

[25] Lindenmayer, A., Mathematical Models for Cellular Interaction in Development I, Filaments with One-Sided Inputs. Journal of Theoritical Biology 1968; **18**: 280-300.